AN ITERATIVE PROCESS TO SOLVE THE GRAPH-COLORING PROBLEM

by

Dennis Spencer Read

United States Naval Postgraduate School

THESIS

AN ITERATIVE PROCESS
TO
SOLVE THE GRAPH-COLORING PROBLEM

by

Dennis Spencer Read

October 1969

An Iterative Process
to
Solve the Graph-Coloring Problem

by

Dennis Spencer Read
Lieutenant Commander, United States Navy
B.S., United States Naval Academy, 1959

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
October 1969

ABSTRACT

The intent of this paper is to describe a method of coloring a map and to present an algorithm for the solution of this problem. A computer program was developed to provide solutions to the problem of coloring a map which consists of a finite number of areas. This algorithm may also be applied to problems other than map-coloring.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

TABLE OF SYMBOLS

SYMBOL	MEANING
M	Number of equations used to describe the border constraints for a map.
N	Number of areas to be colored
x_i	Variable representing the color of area i
A_{MxN}	The MxN matrix defining regions having common borders
$\bar{a}_j$	The $j^{\underline{th}}$ column of A_{MxN}
$\underline{a}_i$	The $i^{\underline{th}}$ row of A_{MxN}
A'_{MxN}	$A_{MxN} + [\bar{a}_j, \ldots, \bar{a}_j]_{1xN}$
β_j	The number of zero elements of each column $\bar{a}_j$ that are not associated with the constraint equations
α_j	The number of zero elements of each column $\bar{a}_j$

I. INTRODUCTION

When coloring a geographical map, it is customary to use different colors for adjacent regions - where adjacent means a common finite boundary. It has been conjectured, but not proved, that "For any subdivision of the plane into non-overlapping regions, it is always possible to mark the regions with one of the numbers 0, 1, 2, 3, in such a way that no two adjacent regions receive the same number."[1]

The statement that a map can be colored with four colors is accredited to Moebius, who first proposed it in 1840. This conjecture has never been proven; however, a satisfactory counter-example has not been demonstrated either. A proof for five-coloring a map may be found in [Courant and Robbins 1941] and [Oystein 1967].

This thesis will use the four-coloring conjecture without proof and present an algorithm that utilizes an iterative routine to determine what color each region should be. A similar iterative technique for general integer programming problems is demonstrated in [Greenberg April 1969] and [Greenberg May 1969], which present the proofs. We are interested in showing whether the iterative technique can be used for large-scale problems such as the map-coloring problem.

The problem formulation suggested by Gomory [Dantzig 1960] is used. Let the map regions be denoted by $i = 1,2,\ldots,N$ and let x_i be an integer valued variable such that

$$0 \leq x_i \leq 3 \tag{1}$$

---

[1] Courant, Richard and Robbins, Herbert; What is Mathematics?, p. 247, Oxford University Press, 1941.

where the four values $x_i = 0, 1, 2,$ or 3 correspond to four different colors. Since any two adjacent areas must have different colors, the constraint may be written in an either/or form

$$\text{either } x_i - x_j \geqslant 1 \qquad \text{or } x_j - x_i \geqslant 1 \tag{2}$$

if i and j have a common border. Equation (2) may be rewritten as:

$$\begin{aligned} x_i - x_j &\geqslant 1 - 4\delta_{ij} \quad (\delta_{ij} = 0,1) \\ x_j - x_i &\geqslant -3 + 4\delta_{ij} \end{aligned} \tag{3}$$

Instead of using (2) and (3) we can simplify the problem by using

$$x_i - x_j \neq 0 \tag{4}$$

where i and j have a common border.

This formulation can be written in matrix form

$$Ax \neq 0 \tag{5}$$

where the elements of A_{MxN}, a_{ij}, are either 0, -1, or 1. We can now enumerate all values of the left side of (5). The column with the least number of zeroes is now selected from A_{MxN}, $\bar{a}_j$,[2] and the value of the corresponding x_j is set equal to one. The column $\bar{a}_j$ is then added column by column to the matrix A_{MxN} to form a new matrix. The process continues in the same way by searching the new matrix for the column with the least number of zeros. In each iteration the corresponding x_j is increased by one. The column values represent the value of the left side of (5). A solution occurs when a column is achieved that has no zeros.

---

[2]The speed with which the algorithm reaches a solution is increased by modifying the method of choosing $\bar{a}_j$. This modification is explained in Section II. A. 4.

The algorithm presented is based upon the four-color conjecture, but is not restricted to the use of only four colors, if more colors are desired.

II. THE COLORING PROCESS

A. THE ALGORITHM

1. Let $x_i = 0$ for $i = 1,\ldots,N$.

2. For every 2 regions of the map with a common border it is required that $x_i - x_j \neq 0$, $i,j, = 1,\ldots,N$; $i \neq j$.

3. Form the M x N matrix (A_{MxN}) defined in step 2, where M is the number of equations required to describe the border constraints and N is the number of areas to be colored. The elements of A_{MxN}, a_{ij}, are either -1, 0, or 1. Note:

   a. For each row, $\underline{a}_i$, $i = 1,\ldots,M$, there are only two non zero elements, -1, and 1, corresponding to the constraints in step 2.

   b. For column $\overline{a}_j$, $j = 1,\ldots,N$, has at least one non zero element and fewer than M non zero elements.

4. Let α_j, $j = 1,\ldots,N$, be the number of zero elements of each $\overline{a}_j$. Let β_j, $j = 1,\ldots,N$, be the number of zero elements of each $\overline{a}_j$ which appear in those positions where $a_{ij} = 0$. Note: $\beta_j \leq \alpha_j$, $j = 1,\ldots,N$. Find the column of A_{MxN} with the smallest β_j. Let j^{**} be the smallest j such that $\beta_{j^{**}} = \min_j \{\beta_j\}$. Let $j^* = j^{**}$.

5. Let $x'_{j^*} = x_{j^*} + 1$. Check all the constraints in which x'_{j^*} appears to insure that the constraint equations (4) are satisfied. If the constraint equations (4) are not satisfied, continue increasing x_{j^*} in increments of one until either (4) is satisfied or $x'_{j^*} = 3$. If $x'_{j^*} = 3$ and (4) is not satisfied, set $x'_{j^*} = 0$ and let j^* be the

smallest $j > j^{**}$ such that $\beta_{j*} = \min_j \{\beta_j\}$ and repeat step 5. If no such j^* exists, then find the smallest α_j and proceed as with the β_j's. If no α_j exists such that x_{j*} and x_i satisfy (4) for all $i \neq j^*$, go to step 9.

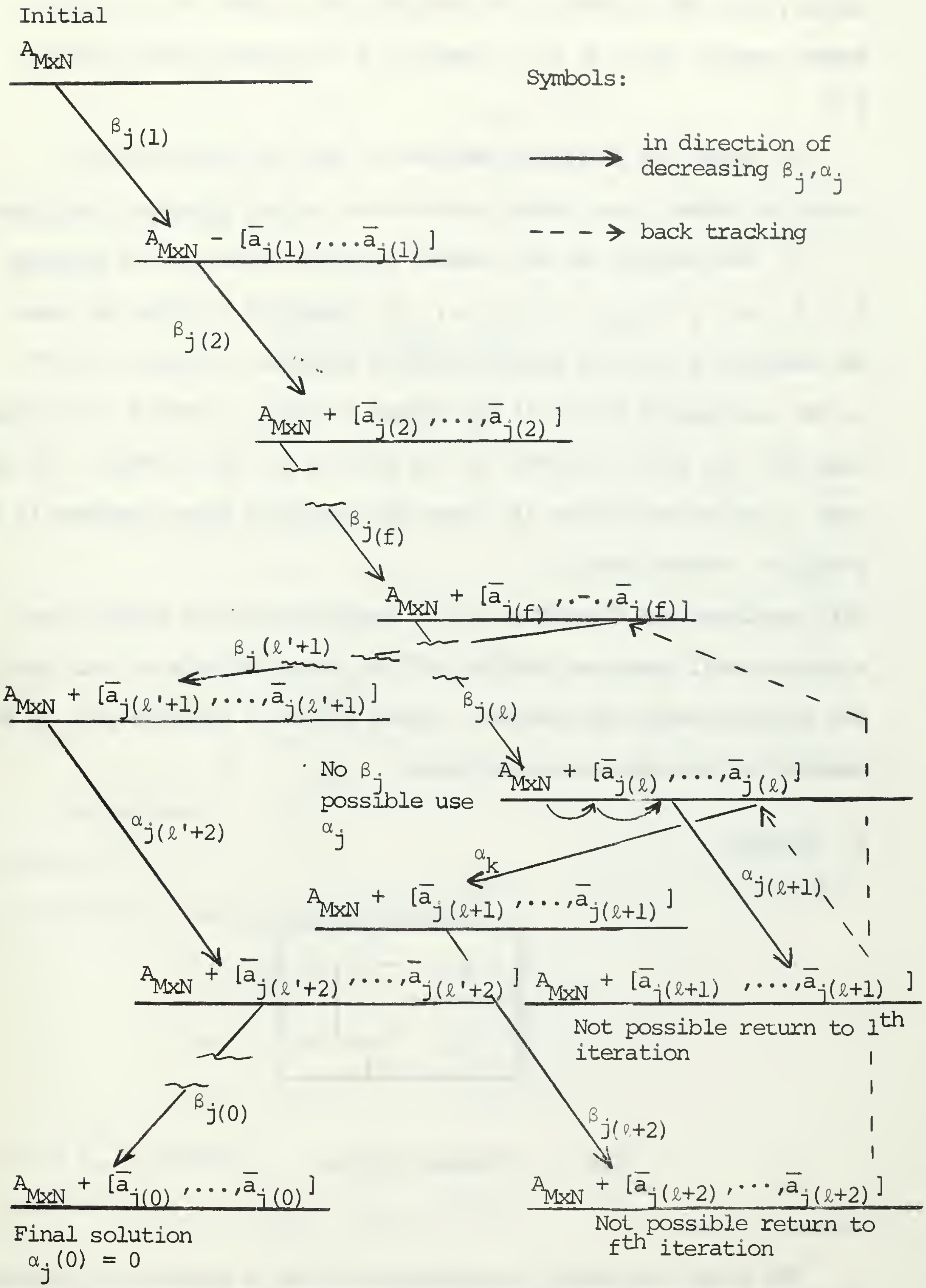
6. Determine the elements of column $\bar{a}_{j*}$ by substituting the presently defined values of x_i, $i = 1,\ldots,N$, into the constraint equations (4). Add the column $\bar{a}_{j*}$, $j = 1,\ldots,N$, to each column of A_{MxN}. Let $A'_{MxN} = A_{MxN} + [\bar{a}_{j*},\ldots,\bar{a}_{j*}]$.

7. Let α_j, $j = 1,\ldots,N$, be the number of zero elements of each $\bar{a}'_j$. Let β_j, $j = 1,\ldots,N$, be the number of zero elements of each $\bar{a}'_j$ which appear in those positions where $a_{ij} = 0$. Let j^{**} be the smallest j such that $\beta_j = \min_j \{\beta_j\}$. Let $j^* = j^{**}$.

8. Continue steps 5 through 8 until $\alpha_j = 0$, in which case a solution has been reached. <u>Note</u>: If $x_j = 3$ at any point during steps 5 through 8, disregard the corresponding column $\bar{a}_j$ in any future iteration, since the corresponding x_j is at its maximum possible value.

9. Let the column $\bar{a}_{j*}$ be the one used to generate the latest A'_{MxN} matrix. For each zero element of $\bar{a}_{j*}$ there are 2 regions, g and h, with a common border where $x_g - x_h = 0$, i.e., a violation of (4). Let F be the set of all such regions g and h. Denote by the subscript f the element of F.

10. For each f, increment the x_f's such that $x_f = x_f + 1$. Check all of the constraints in which x_f appears to see if the constraint equations (4) are satisfied. If for any f, an element of F, the constraint equations (4) are satisfied set $f = j^*$ and go to step 6. If (4) is not satisfied for any i, not an element of F, where regions i and f have a common



Possible Path to a solution to the map-coloring problem

Figure 1

border, note the iteration that defined the present value of x_i. Repeat step 10 until $x_f = 3$. Reset $x_f = 0$. Repeat step 10 for all F.[3]

11. Order the iterations defined in step 10 in decreasing iteration number. Let these iterations be called possible iterations.

12. Reconstruct the most recent possible iteration and examine $\beta_j = \beta_{j*}$ or $\alpha_j = \alpha_{j*}$, $j = j* + 1,\ldots,N$, depending on which was used to determine x_{j*} in the iteration being examined. Choose a new j* in the same manner as the j* was chosen in step 5. When a j* is found such that x_{j*} and x_i satisfy (4) for all $i \neq j*$, go to step 6. If no such j* exists satisfying (4), then the iteration being examined is not possible. Repeat step 12.

This completes the algorithm. If no possible solution exists, the algorithm will continue "backing up" the generated matrix chain until the original matrix is reached. Figure 1 shows a possible path to the solution of the map-coloring problem.

B. EXAMPLE

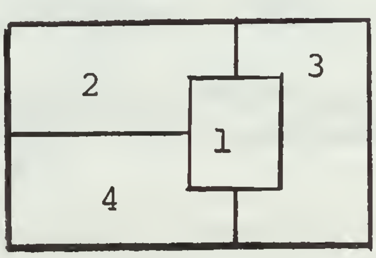


Fig. 2. Example Problem

[3]The speed with which the algorithm reaches a solution is increased by starting the backtracking procedure when no variable associated with the minimum β_j or α_j can be set at any integer value between 1 and 3.

where

$$x_1 - x_2 \neq 0 \qquad x_2 - x_3 \neq 0$$
$$x_1 - x_3 \neq 0 \qquad x_2 - x_4 \neq 0$$
$$x_1 - x_4 \neq 0 \qquad x_3 - x_4 \neq 0$$

set $x_i = 0$ for all i

we write

	x_1	$-x_2$	x_3	x_4
	1	-1	0	0
	1	0	-1	0
	1	0	0	-1
	0	1	-1	0
	0	1	0	-1
	0	0	1	-1
α_j	3	3	3	3
β_j	3	3	3	3
column used	*			

then $x_1 = 1$

the column to add $(\bar{a}_{j*})$ is defined by:

$$x_1 - x_2 = 1 \qquad x_2 - x_3 = 0$$
$$x_1 - x_3 = 1 \qquad x_2 - x_4 = 0$$
$$x_1 - x_4 = 1 \qquad x_3 - x_4 = 0$$

where $\bar{a}_{j*}^T = [111000]$

then adding $[\bar{a}_{j*},\ldots,\bar{a}_{j*}] + A_{MxN}$ we write

	x_1	x_2	x_3	x_4
	2	0	1	1
	2	1	0	1
	2	1	1	0
	0	1	-1	0
	0	1	0	-1
	0	0	1	-1
α_j	3	2	2	2
β_j	3	1	1	1
column used		*		

then $x_2 = 2$

the column to add $(\bar{a}_{j*})$ is defined by:

$$x_1 - x_2 = -1 \qquad x_2 - x_3 = 2$$
$$x_1 - x_3 = 1 \qquad x_2 - x_4 = 2$$
$$x_1 - x_4 = 1 \qquad x_3 - x_4 = 0$$

where $\bar{a}_{j*}{}^T = [-111220]$

then adding $[\bar{a}_{j*}, \ldots, \bar{a}_{j*}] + A_{MxN}$ we write

	x_1	x_1	x_3	x_4
	0	-2	-1	-1
	2	1	0	1
	2	1	1	0
	2	3	1	2
	2	3	2	1
	0	0	1	-1
α_j	2	1	1	1
β_j	1	1	0	0
column used			*	

then $x_3 = 3$

the column to add $(\bar{a}_{j*})$ is defined by:

$$x_1 - x_2 = -1 \qquad x_2 - x_3 = -1$$

$$x_1 - x_3 = -2 \qquad x_2 - x_4 = 2$$

$$x_1 - x_4 = 1 \qquad x_3 - x_4 = 3$$

where $\bar{a}_{j*}{}^T = [-1\ -2\ 1\ -1\ 2\ 3]$

then adding $[\bar{a}_{j*},\ldots,\bar{a}_{j*}] + A_{MxN}$ we write

	x_1	x_2	x_3	x_4
	0	-2	-1	
	-1	-2	-3	
	2	1	1	
	-1	0	-2	
	2	3	2	
	3	3	4	
α_j	1	1	0	
			*	

there is no need to calculate column $\bar{a}'_4$ because there are no zeros in column $\bar{a}'_3$ and a solution has been reached. The solution is:

Area number	Color number
1	1
2	2
3	3
4	0

III. CONCLUSION

The algorithm described will solve the map-coloring problem. Although only four colors were used in the example in Section II-B, the algorithm can be adapted to solve a problem for any feasible number of colors. Several representative map-coloring problems were solved using the enclosed computer program. Appendix A shows a solution to four-coloring the continental United States. In the representative problems solved, both four and five colors were used.

If the solution is not feasible, e.g., the use of four colors when five colors are required, as in the split-state problem, the algorithm may consume several hours of computer time before showing that no solution exists. The computer time is consumed in back-tracking the solution path and in investigating every possible solution path for both the α's and the β's.

APPENDIX A

SAMPLE PROBLEM SOLUTION

The solution for four-coloring the continental United States of America, including boundaries and major water bodies, is as follows:

STATE	COLOR
Maine	0
New Hampshire	3
Vermont	0
Massachusetts	2
Rhode Island	3
Connecticut	0
New York	3
New Jersey	0
Pennsylvania	2
Delaware	3
Maryland	0
Virginia	2
North Carolina	3
South Carolina	0
Georgia	2
Florida	0
Alabama	3
Tennessee	1
West Virginia	1
Ohio	0
Indiana	2
Kentucky	3
Illinois	0
Wisconsin	2
Michigan	3
Minnesota	0
Iowa	3
Missouri	2
Arkansas	3
Mississippi	2
Louisiana	0
Texas	2
Oklahoma	0
Kansas	3
Nebraska	0
South Dakota	1
North Dakota	3
Montana	0
Wyoming	2

Colorado	1
New Mexico	3
Arizona	1
Utah	3
Idaho	1
Washington	0
Oregon	2
Nevada	0
California	3

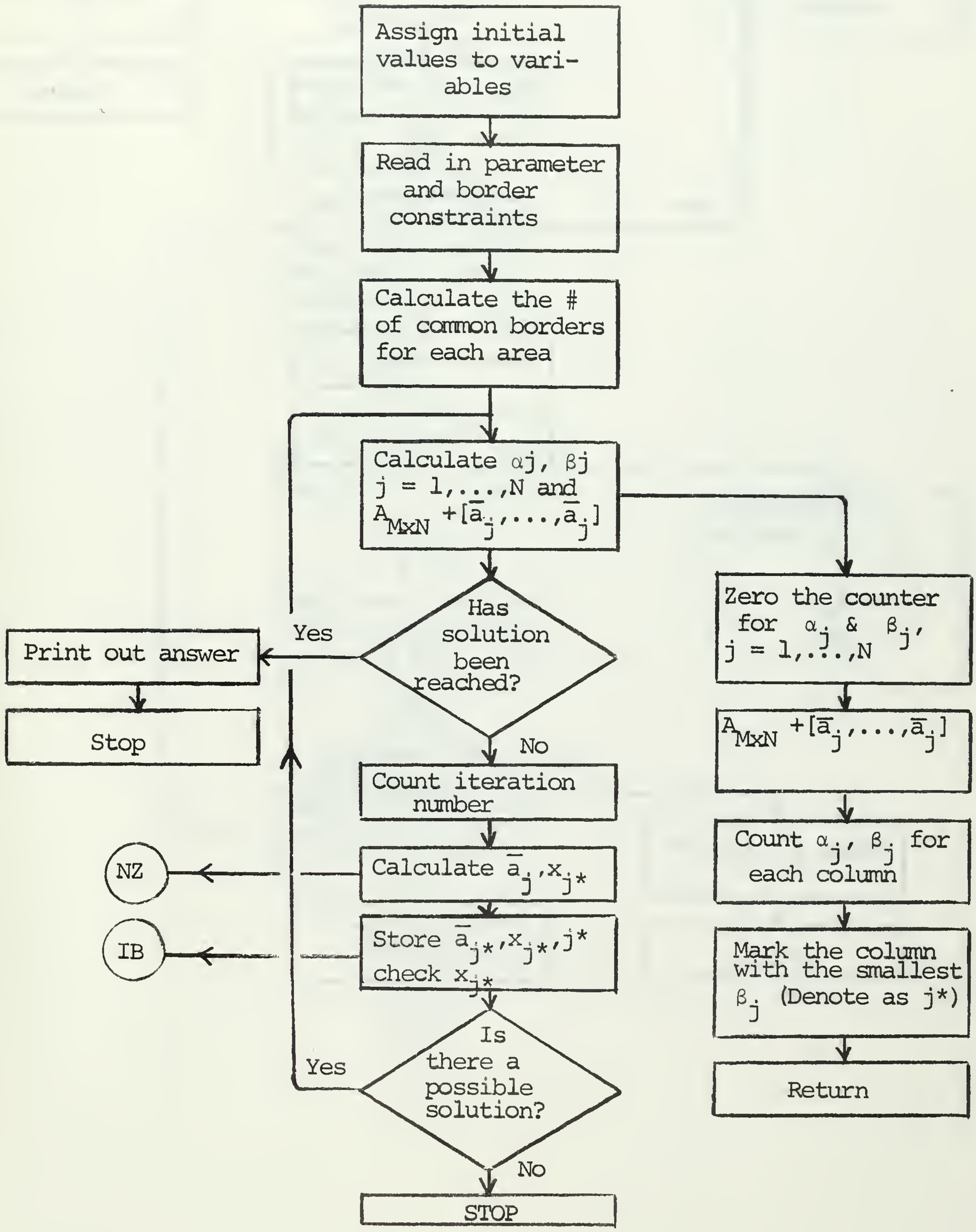
BORDER AREAS

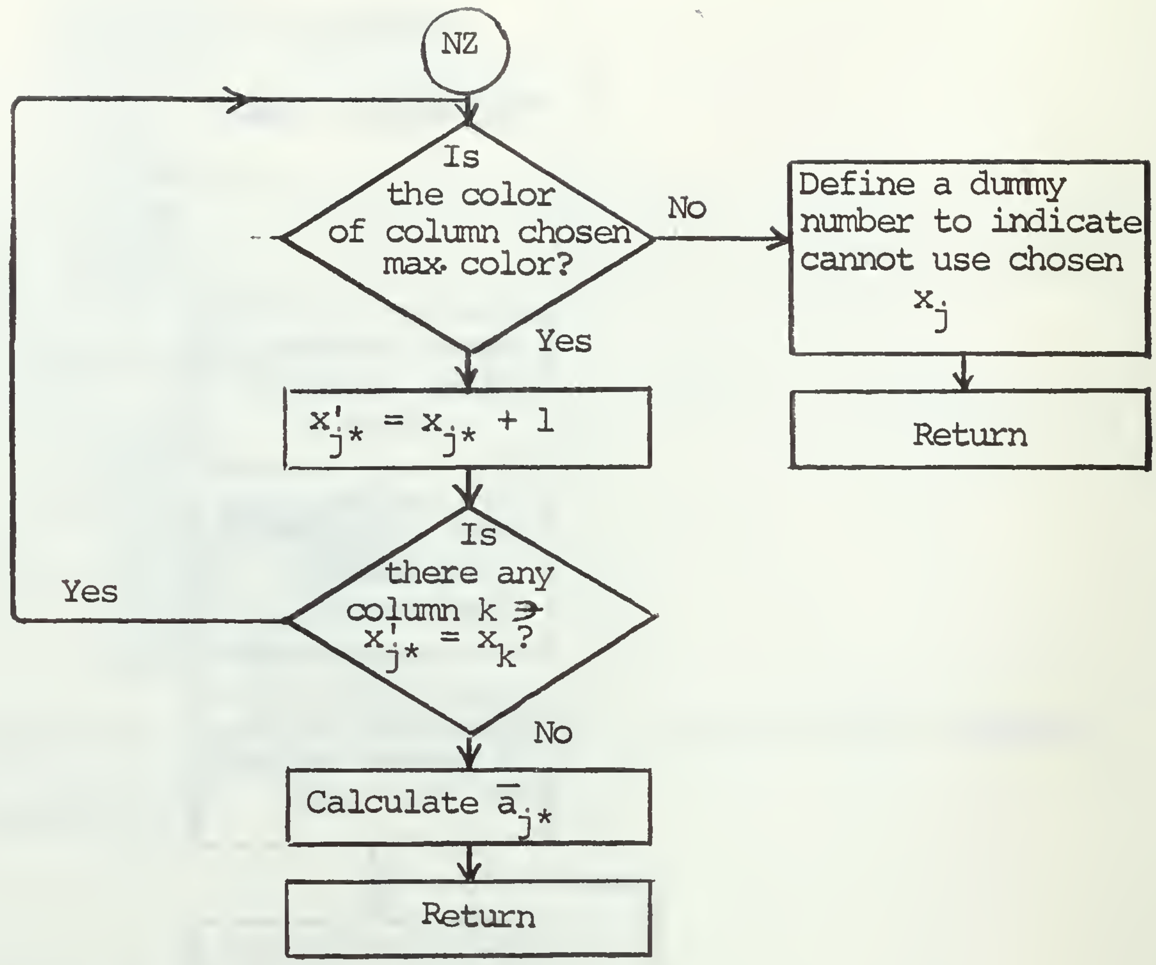
Canada	2
Mexico	0
Water - including:	1
Atlantic Ocean	
Pacific Ocean	
Great Lakes	
Great Salt Lake	

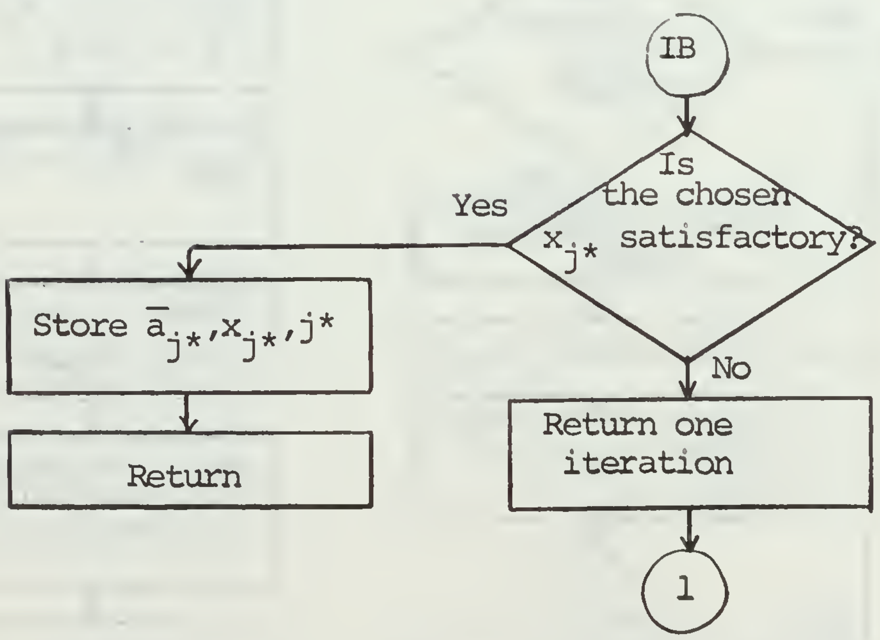
The above problem was solved in 97.2 seconds on the IBM 360/67 computer and used 81 iterations.

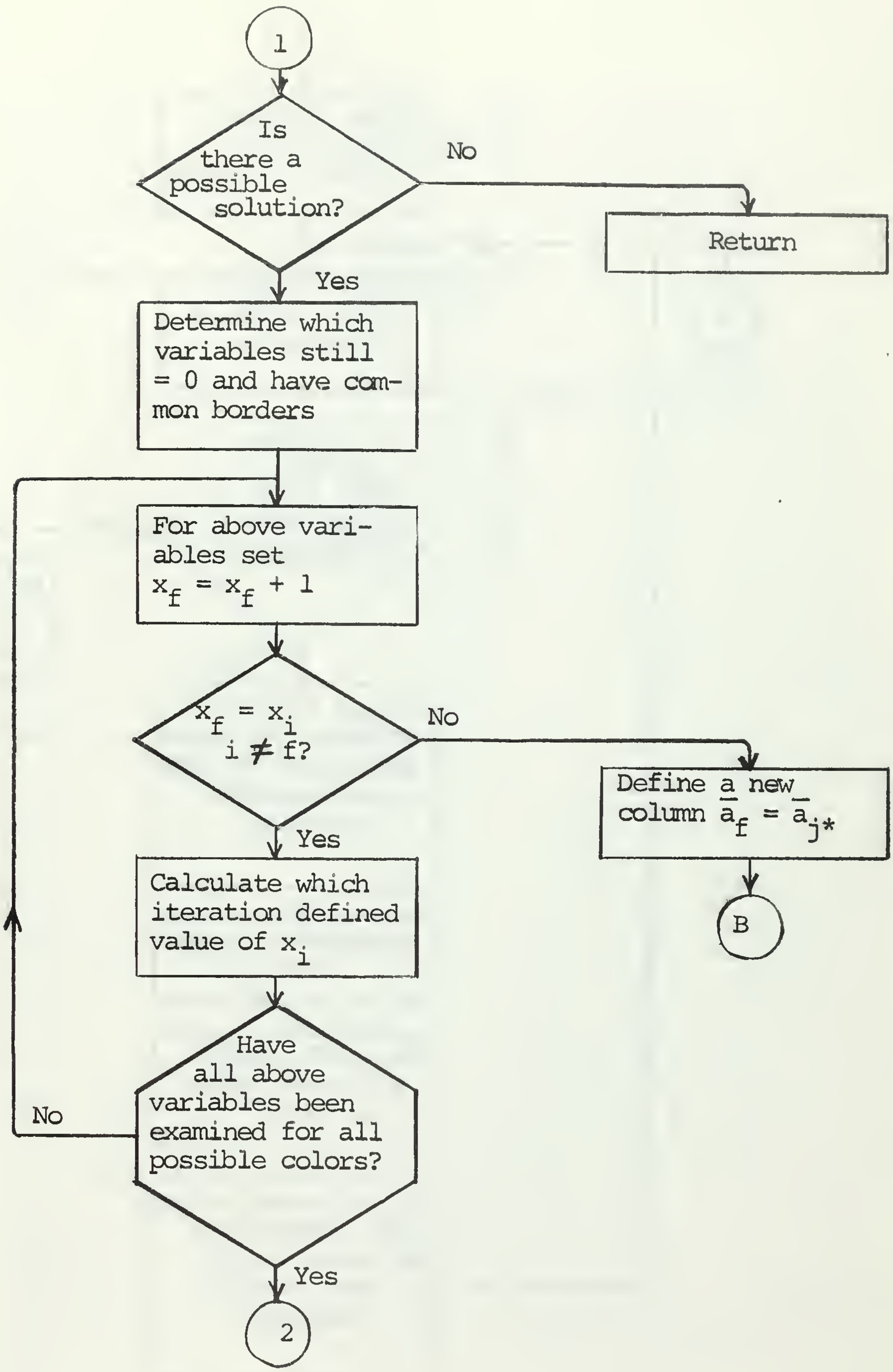
APPENDIX B

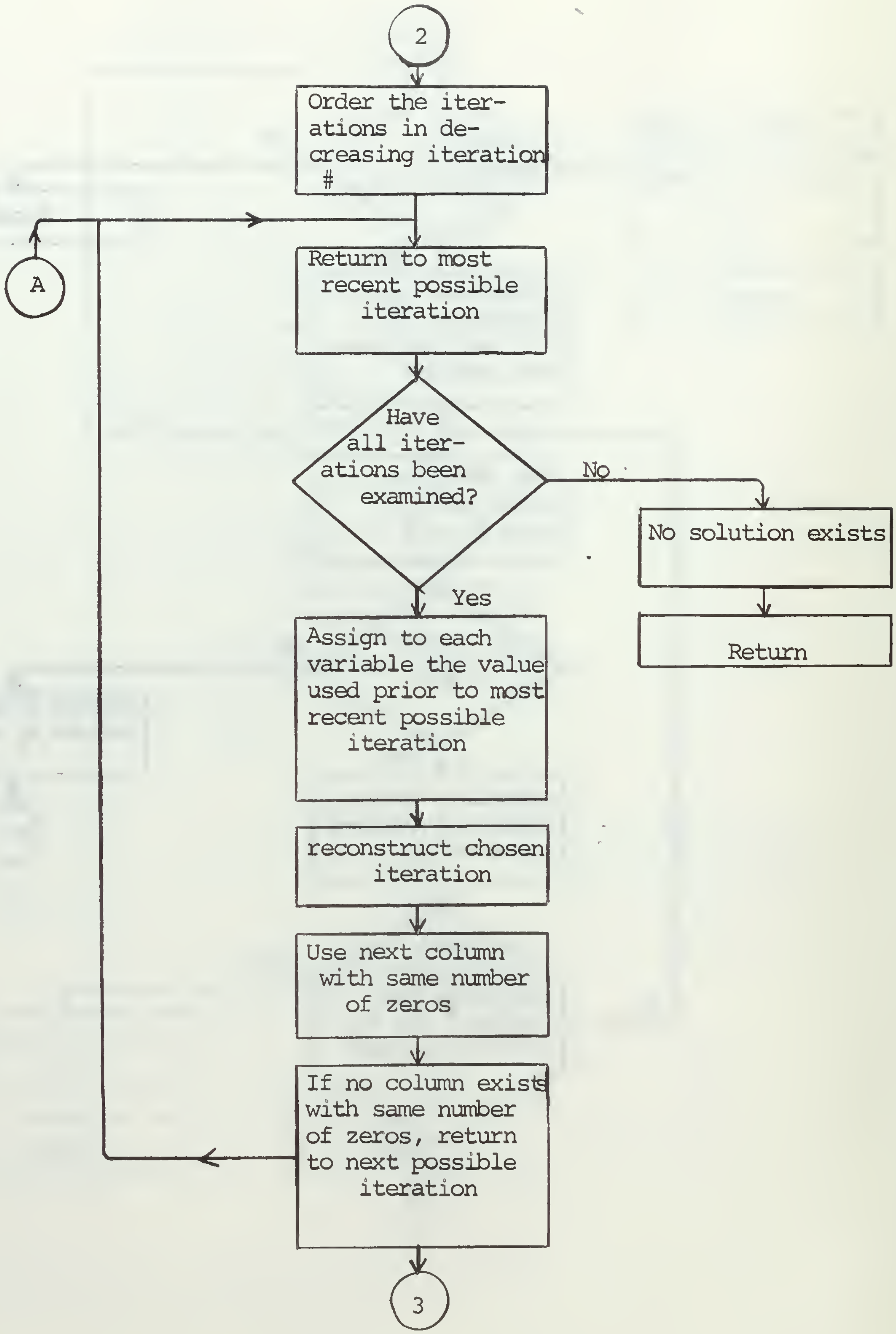
PROGRAM FLOW CHART

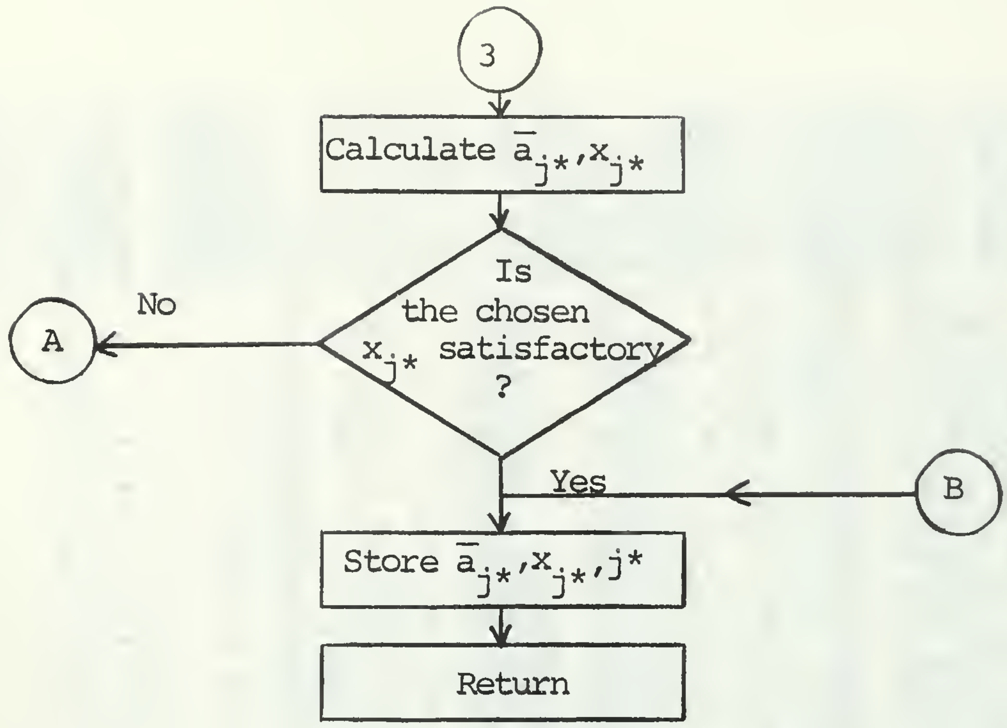



NZ
Is the color of column chosen max. color?
No
Define a dummy number to indicate cannot use chosen x_j
Return
Yes
$x'_{j*} = x_{j*} + 1$
Is there any column k ∋ $x'_{j*} = x_k$?
Yes
No
Calculate $\bar{a}_{j*}$
Return


IB
Is the chosen x_{j*} satisfactory?
Yes
Store $\bar{a}_{j*}, x_{j*}, j*$
Return
No
Return one iteration
1


1
Is there a possible solution?
No
Return
Yes
Determine which variables still = 0 and have common borders
For above variables set $x_f = x_f + 1$
$x_f = x_i$ $i \neq f$?
No
Define a new column $\bar{a}_f = \bar{a}_{j*}$
B
Yes
Calculate which iteration defined value of x_i
Have all above variables been examined for all possible colors?
No
Yes
2


2
Order the iterations in decreasing iteration #
A
Return to most recent possible iteration
Have all iterations been examined?
No
No solution exists
Return
Yes
Assign to each variable the value used prior to most recent possible iteration
reconstruct chosen iteration
Use next column with same number of zeros
If no column exists with same number of zeros, return to next possible iteration
3


3
Calculate $\bar{a}_{j*}, x_{j*}$
Is the chosen x_{j*} satisfactory ?
No
A
Yes
B
Store $\bar{a}_{j*}, x_{j*}, j*$
Return

```
C ITERATIVE ROUTINE TO FIND A SOLUTION TO THE MAP COLORING PROBLEM
C
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C    BECAUSE OF THE METHOD USED IN DEFINING THE VALUE OF THE VARIABLE
C    AND THE COLUMN TO ADD TO THE INITIAL MATRIX, THE VALUE OF THE LAST VARIABLE
C    MAY NOT BE CORRECT IN THE FINAL PRINTOUT.  FOR THIS REASON WHEN A
C    COLUMN WITH NO ZEROS IS GENERATED THE PROGRAM WILL PRINT OUT 'LAST VARIABLE
C    USED IS'.  THE VALUE OF THIS VARIABLE CAN THEN EASILY BE CALCULATED AS
C    PRINTED VALUE+1 IF NECESSARY.
C    THE PROGRAM USES 37,000 BYTES OF OBJECT CODE ON AN IBM360/67, USING
C    FORTRAN IV G LANGUAGE, PLUS (5*(N*NN)+2*N+19*NN)*4 BYTES OF ARRAY AREA
C    PLUS REQUIRED BUFFER FOR STORAGE.
C    DIMENSION CARDS IN THE MAIN PROGRAM AND IN SUBROUTINE IBAK MUST BE
C    CHANGED FOR ANY CHANGE IN THE DATA.
C * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
C
C   SYMBOLS ARE:
C    ITAB: THE VALUE OF THE INITIAL N BY NN MATRIX
C    IADD: COMPONENTS OF COLUMN WITH THE LEAST NUMBER OF ZEROS
C    MAT:  THE N BY NN MATRIX THAT = ITAB PLUS IADD
C    DUM: DUMMY VALUE TO SIGNIFY COMPLETION OF THE ALGORITHM
C    KNT:  COUNT TO INDICATE HOW MANY VALUES OF THE VARIABLES HAVE BEEN DEFINED
C    N AND NN ARE THE NUMBER OF EQUATIONS AND VARIABLES RESPECTFULLY
C    MV: THE VALUE OF THE VARIABLE
C    KLR:  THE MAXIMUM NUMBER OF COLORS TO USE (MAX NO.=KLR+1)
C    ITSV: DENOTES A VARIABLE THAT IS AT ITS  MAXIMUM VALUE
C    KOLM: DEFINES THE TOTAL NUMBER OF AREAS BORDERING ON ANY GIVEN AREA
C    BA AND E : ARE DUMMYS TO 'MARK' A POINT WHERE A CHANGE IN ROUTINE IS
C               NECESSARY
C    NP3 AND NNX3:   DIMENSION THE SIZE OF MATRICES USED FOR BACKTRACKING
C
C  READ IN MATRIX SIZE AND MAXIMUM NUMBER OF COLORS TO BE USED
      READ(5,100) N,NN,KLR                                              MAIN0010
  100 FORMAT(3I5)                                                       MAIN0020
C
      DIMENSION ITAB(151,51),MV(51),IADD(151),MAT(151,51),ITSV(51)      MAIN0030
      DIMENSION NZ(51),KOLM(51),NZCON(51)                               MAIN0031
C THE DIMENSIONS ON ABOVE VARIABLES MUST BE CHANGED FOR EVERY PROGRAM CHANGE
C
C  ZERO ALL THE VARIABLES AND COUNTERS.
      DUM=0.0                                                           MAIN0040
      KNT=0                                                             MAIN0041
      E=0.0                                                             MAIN0042
      BA=0.0                                                            MAIN0043
      NP3=N+3                                                           MAIN0044
      NNX3=NN*3                                                         MAIN0045
      DO 110 I=1,NN                                                     MAIN0050
      KOLM(I)=0                                                         MAIN0057
```

```
      ITSV(I)=C                                                         MAIN0058
  110 MV(I)=0                                                           MAIN0060
      DO 115 I=1,N                                                      MAIN0080
  115 IADD(I)=C                                                         MAIN0090
      DO 117 I=1,N                                                      MAIN0095
      DO 117 J=1,NN                                                     MAIN0096
  117 ITAB(I,J)=0                                                       MAIN0097
C  READ IN THE INITIAL VALUES DEFINING BORDER AREAS
  119 READ(5,120) I,J,K,ITAB(I,J),ITAB(I,K),IDUM                        MAIN0100
  120 FORMAT(6I5)                                                       MAIN0110
      IF(IDUM.EQ.0) GO TO 119                                           MAINC111
C  DETERMINE THE NUMBER OF BORDERING  AREAS FOR EACH STATE
      DO 121 J=1,NN                                                     MAIN0115
      DO 121 I=1,N                                                      MAIN0116
      IJ=ITAB(I,J)                                                      MAIN0117
  121 KCLM(J)=KOLM(J)+IABS(IJ)                                          MAIN0118
C  DETERMINE THE COLUMN TO USE FOR THE NEXT ITERATION AND MARK IT
  125 CALL NMATX(NN,N,ITAB,IADD,MAT,II,DUM,NZ,ITSV,NZCON)               MAINC120
C  IF SOLUTION HAS BEEN REACHED, PRINT OUT THE ANSWER
      IF(DUM.GT.0.0) GO TO 130                                          MAIN0130
C  INCREASE COUNTER BY ONE
      KNT=KNT+1                                                         MAIN0142
C  DETERMINE THE NEW VALUE OF THE VARIABLE ,   INDICATE IF SOLUTION IS
C  INFEASIBLE SO THAT A NEW PATH CAN BE TAKEN, AND DEFINE THE NEW COLUMN
C   TO BE ADDED TO THE INITIAL TABLEAU
  140 CALL NZERO(N,NN,II,ITAB,ITSV,MV,E,KLR,IADD,BA,KOLM)               MAIN0150
      IF(BA.NE.0.0) GO TO 140                                           MAIN0153
C  SAVE THE COLUMN ADDED TO THE INITIAL TABLEAU/VARIABLE USED.  IF SOLUTION NOT
C  FEASIBLE, GO BACK TO FIRST FEASIBLE TABLEAU AND PICK A NEW VARIABLE TO USE
      CALL IBACK(N,NN,IADD,KNT,II,NZ,ITAB,ITSV,DUM,MV,MAT,E,KLR,KOLM,   MAIN0155
     1NZCON,NP3,NNX3)                                                   MAIN0156
      IF(DUM.GT.0.0) GO TO 130                                          MAIN0158
      GO TO 125                                                         MAIN0160
C  WRITE OUT COLORS FOR EACH AREA.
  130 DO 135 I=1,NN                                                     MAIN0170
      WRITE(6,150) I,MV(I)                                              MAIN0180
  150 FORMAT(10X,'FOR AREA NUMBER     USE COLOR NUMBER'T27,I3,T48,I3)   MAIN0190
  135 CONTINUE                                                          MAIN0200
      STOP                                                              MAIN0210
      END

      SUBROUTINE NMATX(NN,N,IM,IT,MT,II,D,NZ,ISV,NZCON)                 NMAT0010
C  SUBPROGRAM TO DO THE FOLLOWING:
C  (1) ADD COLUMN WITH LEAST NUMBER OF ZEROS  TO INITIAL MATRIX
C  (2) IF ANY COLUMN HAS NO ZEROS STOP THE ITERATION
C     SYMBOLS ARE:
C     MT=MAT; IM=ITAB,D=DUM,  IT=IADD,    ISV=ITSV
```

```
C     II: THE SUBSCRIPT OF THE COLUMN WITH LEAST NUMBER OF ZEROS
C     NZ:THE NUMBER OF ZEROS IN EACH COLUMN
C     NZCON: COUNTS THE NUMBER OF ZEROS IN A COLUMN THAT ARE NOT DEFINED IN THE
C            BORDER EQUATIONS
C
      DIMENSION MT(N,NN),IM(N,NN),IT(N),NZ(NN),ISV(NN),NZCON(NN)         NMAT0020
C   ZERO THE VALUE OF THE NUMBER OF ZEROS IN EACH COLUMN
      DO 320 I=1,NN                                                      NMAT0023
      NZCON(I)=0                                                         NMAT0024
  320 NZ(I)=0                                                            NMAT0025
C   ADD THE DESIGNATED COLUMN TO THE INITIAL MATRIX AND COUNT THE NUMBER OF
C    ZEROS IN EACH COLUMN
      DO 300 J=1,NN                                                      NMAT0030
      DO 305 K=1,N                                                       NMAT0040
      MT(K,J)=IM(K,J)+IT(K)                                              NMAT0050
      IF(MT(K,J).EQ.0) CALL KONFLT(NN,J,K,NZ,NZCON,IM,N)                 NMAT0060
  305 CONTINUE                                                           NMAT0065
C   IF ANY COLUMN HAS NO ZEROS, A SOLUTION HAS BEEN REACHED
      IF(NZ(J).EQ.0) GO TO 350                                           NMAT0080
C   IF VARIABLE IS AT MAXIMUM VALUE, DEFINE NUMBER OF ZEROS TO BE AT MAXIMUM SO
C    THAT COLUMN WILL NOT BE FURTHER CONSIDERED
      IF(ISV(J).EQ.N) NZ(J)=N                                            NMAT0085
  300 CONTINUE                                                           NMAT0090
C   FIND COLUMN WITH LEAST NUMBER OF ZEROS
      CALL IVAR(NN,NZ,II,NZCON,N)                                        NMAT0100
      RETURN                                                             NMAT0130
C   A SOLUTION HAS BEEN REACHED
  350 D=NN                                                               NMAT0140
      II=J                                                               NMAT0142
      WRITE(6,390) II                                                    NMAT0144
  390 FORMAT(//10X,'LAST VARIABLE USED IS'T35,I3)                        NMATC145
      RETURN                                                             NMAT0150
      END                                                                NMAT0160


      SUBROUTINE IVAR(NN,MZ,II,NC,N)                                     IVAR0010
C   SUBROUTINE TO DETERMINE WHICH COLUMN HAS THE LEAST NUMBER OF ZEROS THAT
C      ARE NOT ASSOCIATED WITH BORDER AREAS.
C     SYMBOLS ARE:
C     MZ=NZ;  NC=NZCON
C     NP: DUMMY TO INSURE ALL COLUMNS ARE CHECKED
C     II: DENOTES THE COLUMN(VARIABLE) TO BE USED.
C
      DIMENSION MZ(NN),NC(NN)                                            IVAR0020
C   ZERO VALUES
      II=1                                                               IVAR0030
      NP=NN-1                                                            IVAR0035
C   IF AN AREA IS AT ITS MAXIMUM COLOR, INSURE IT IS NOT CHOSEN
```

```
      DO 210 I=1,NN                                                     IVAR0037
  210 IF(MZ(I).EQ.N) NC(I)=N                                            IVAR0038
C  PICK THE COLUMN WITH THE LEAST NUMBER OF NONBORDERING ZERO AREAS
      DO 200 K=1,NP                                                     IVAR0040
  200 IF(NC(K+1).LT.NC(II)) II=K+1                                      IVAR0050
      RETURN                                                            IVAR0060
      END                                                               IVAR0070

      SUBROUTINE NZERO(N,NN,II,IT,ITSV,MV,E,KL,IA,BA,KM)                NZER0010
C    A SUBPROGRAM TO DETERMINE WHICH EQUATIONS CORRESPOND TO THE VARIABLE
C     CHOSEN. // SYMBOLS ARE AS DEFINED IN MAIN PROGRAM; OTHER SYMBOLS ARE:
C     ITSV:  DENOTES VARIABLE IS AT MAXIMUM VALUE
C     BA: DUMMY TO INDICATE THAT 2  ADJACENT AREAS HAVE THE SAME COLOR
C     K2: COUNTER TO INDICATE WHEN ALL BORDER AREAS HAVE BEEN EXAMINED
C     E: A DUMMY TO INDICATE THAT TWO VARIABLES ARE EQUAL AND AT MAX
C     OTHER SYMBOLS: IT=ITAB; KL=KLR
C
      DIMENSION IT(N,NN),ITSV(NN),MV(NN),IA(N),KM(NN)                   NZER0020
C  ZERO THE COUNTERS AND DUMMIES
      BA=0.0                                                            NZER0024
      E=0.0                                                             NZER0025
      K2=0                                                              NZER0026
C   IF COLUMN CHOSEN IS AT MAXIMUM VALUE, RETURN AND CHOOSE A NEW COLUMN
      IF(MV(II).EQ.KL) GO TO 610                                        NZER0035
C   INCREASE VALUE OF VARIABLE BY ONE
      MV(II)=MV(II)+1                                                   NZER0036
C   IF COLOR IS AT THE MAXIMUM VALUE, MAKE A NOTE OF THE VARIABLE
      IF(MV(II).EQ.KL) ITSV(II)=N                                       NZER0037
C  CHECK EACH ROW FOR VARIABLE USED TO CHECK THE COLOR OF THE ADJOINING AREAS
      DO 600 JJ=1,N                                                     NZER0038
      IF(IT(JJ,II).NE.0) CALL INFEAS(N,NN,MV,IT,II,JJ,KL,IA,BA,KM,K2)   NZER0040
      IF(BA.NE.0.0) RETURN                                              NZER0045
  600 CONTINUE                                                          NZER0050
      RETURN                                                            NZER0060
  610 E=N                                                               NZER0064
      RETURN                                                            NZER0065
      END                                                               NZER0070

      SUBROUTINE IBACK(N,NN,IA,KNT,II,NZ,IN,ITSV,D,MV,MT,E,KL,KM,NC,N3, IBAK0010
     1NX3)                                                              IBAK0011
C SUBPROGRAM TO DO THE FOLLOWING:
C  (1)SAVE THE COLUMN ADDED TO THE INITIAL MATRIX
C  (2) SAVE THE INDEX VALUE OF THE COLUMN ADDED TO THE INITIAL MATRIX AND SAVE
C       THE INDICATOR FROM WHERE THE COLUMN CAME(ALPHA OR BETA)AND SAVE THE
C       COLOR OF THE VARIABLE
C  (3) REINDEX THE COLUMN, IF NO OTHER COLUMN CAN BE CHOSEN
```

```
C    SYMBOLS ARE:
C   MT= MAT;  IA=IADD;   IN=ITAB   NC=NZCON
C   AD- DUMMY VARIABLE TO INDICATE COLUMN USED HAD THE LEAST NUMBER OF ZEROS IN
C       THE TABLEAU
C   BA, E,KNT,II,NZ,AND ITSV SAME AS PREVIOUSLY DEFINED
C   KOLSV- THE MATRIX THAT SAVES THE VALUE OF THE COLUMN ADDED,VARIABLE USED,
C          NUMBER OF ZEROS IN THE COLUMN, AND COLOR OF VARIABLE
C   IITMP:  SAVES THE COLUMN NUMBER OF VARIABLES THAT ARE ADJACENT AND AT THE
C           ZERO LEVEL
C   MVMCH: SAVES THE COLUMN NUMBER OF AREAS ADJACENT TO COLUMNS DETERMINED IN
C          IITEMP,IF THE VARIABLES ASSOCIATED WITH IITEMP WERE INCREASED TO
C          MAX COLOR
C   ITER:  INDICATES IN WHICH ITERATION THE COLOR MATCHES WOULD OCCUR
C   L4: INDICATE WHICH ITERATION MUST BE RETURNED TO
C   L2: COUNTS THE NUMBER OF VARIABLES ASSOCIATED WITH IITMP
C   KN1: COUNTS THE NUMBER OF VARIABLES ASSOCIATED WITH MVMCH
C   KN2: DUMMY TO INDICATE NO CHANGE IN KN1
C   IND:   INDICATES WHEN NO OTHER COLUMN FROM  NONBORDERING AREAS CAN BE
C          EXAMINED
C
      DIMENSION IITMP(51),MVMCH(51),ITER(151),KOLSV(154,153),NU(153)     IBAK0020
C THE DIMENSIONS ON ABOVE VARIABLES MUST BE CHANGED FOR EVERY PROGRAM CHANGE
C  DIMENSIONS ON KOLSV ARE (N+3,NN*3) AND ON NU ARE(NN*3)
C
      DIMENSION NZ(NN),IA(N),IN(N,NN),ITSV(NN),MV(NN),MT(N,NN),KM(NN)    IBAK0016
      DIMENSION NC(NN)                                                  IBAK0017
C  ZERO THE DUMMIES AND THE INDICATOR
      IND=0                                                             IBAK0020
      AD=0.0                                                            IBAK0021
      BA=0.0                                                            IBAK0022
      KOLSV(N+2,KNT)=IND                                                IBAK0023
      NU(KNT)=0                                                         IBAK0024
C  IF COLOR CHOSEN IS AT MAXIMUM VALUE, GO TO ANOTHER COLUMN
      IF(E.NE.0.0)CALL KLRMCH(N,NN,II,AD,MV,NZ,IN,ITSV,IA,E,MT,KL,KOLSV,IBAK0025
     1KNT,KM,NC,IND,NU,N3,NX3)                                          IBAK0026
C  IF NO OTHER COLUMN CAN BE USED, RETURN TO PREVIOUS ITERATION
      IF(AD.NE.0.0) GO TO 410                                           IBAK0028
C  SAVE NECESSARY VALUES TO BE USED IN RECONSTRUCTING THIS ITERATION IF REQUIRED
      DO 400 I=1,N                                                      IBAK0030
  400 KOLSV(I,KNT)=IA(I)                                                IBAK0040
      KOLSV(N+1,KNT)=II                                                 IBAK0050
      KOLSV(N+3,KNT)=MV(II)                                             IBAK0063
      RETURN                                                            IBAK0070
C
C  SUBROUTINE NOW PROCEEDS TO RETURN TO THE LAST FEASIBLE ITERATION USED. THE
C   VALUE OF THE VARIABLE PREVIOUSLY USED IS ERASED, AND THE ROUTINE MOVES OVER
C   TO THE NEXT COLUMN WITH THE SAME NUMBER OF ZEROS AND ATTEMPTS TO USE THAT
C   COLUMN AS THE VARIABLE
```

```
C
C  RETURN ONE ITERATION
 410 KNT=KNT-1                                                           IBAK0100
C  IF ALL ITERATIONS HAVE BEEN EXAMINED, NO SOLUTION EXISTS
      IF(KNT.EQ.0) GO TO 450                                             IBAK0110
C  IF THE VARIABLE WAS AT MAXIMUM VALUE, REZERO THE INDICATOR
      IF(MV(II).EQ.KL) ITSV(II)=0                                        IBAK0090
C  ZERO THE COUNTERS AND INDICATORS
      L4=0                                                               IBAK0091
      L2=0                                                               IBAK0092
      KN1=0                                                              IBAK0093
C  DETERMINE THE COLUMNS ASSOCIATED WITH THE ZERO VALUES IN IADD
      DO 482 I=1,N                                                       IBAK0094
 482 IF(IA(I).EQ.0) CALL KNTVAR(N,NN,IN,I,IITMP,L2)                      IBAK0095
C
C  IF THE COLOR OF THE ADJACENT AREAS THAT ARE PRESENTLY AT A ZERO COLOR LEVEL
C   WERE INCREASED TO THE MAXIMUM COLOR, DETERMINE WHAT AREAS WOULD THEN HAVE
C   MATCHING COLORS
C
      DO 460 J=1,L2                                                      IBAK0096
C  ASSIGN A TEMPORARY VALUE TO VARIABLE THAT IS AT ZERO LEVEL
      IIS=IITMP(J)                                                       IBAK0097
C  ASSIGN AN INCREASING COLOR VALUE TO ZERO LEVEL VARIABLES
      DO 461 I=1,KL                                                      IBAK0098
      MV(IIS)=I                                                          IBAK0099
C  ASSIGN INITIAL VALUE TO AN INDICATOR TO SHOW WHEN A COLOR ASSIGNED TO AN
C   AREA IS NOT THE SAME AS SOME BORDERING AREA
      KN2=KN1                                                            IBAK0100
C  CHECK EACH ROW FOR THE COLOR USED TO CHECK THE COLOR OF THE ADJOINING AREAS.
      DO 462 JJ=1,N                                                      IBAK0101
 462 IF(IN(JJ,IIS).NE.0) CALL LKINF(N,NN,IIS,KN1,IN,MV,MVMCH,JJ)         IBAK0102
C  IF PRESENT COLOR DOES NOT MATCH WITH ANY BORDER AREAS, STOP THE EXAMINING
C  PROCESS AND RETURN TO MAIN PROGRAM.
      IF(KN2.EQ.KN1) CALL NOMCH(N,NN,IIS,II,I,MV,IA,IN,NU,KNT,KL,ITSV,  IBAK0103
     1NX3)                                                               IBAK0103
      IF(KN2.EQ.KN1) GO TO 441                                           IBAK0104
 461 CONTINUE                                                            IBAK0105
C  REZERO VARIABLE EXAMINED
      MV(IIS)=0                                                          IBAK0106
 460 CONTINUE                                                            IBAK0107
C  DETERMINE FROM WHICH ITERATION EACH OF THE VARIABLES PICKED ABOVE CAME
      DO 471 J=1,KN1                                                     IBAK0108
      ITER(J)=0                                                          IBAK0109
      DO 471 I=1,KNT                                                     IBAK0110
      IF(MVMCH(J).EQ.KOLSV(N+1,I)) ITER(J)=I                             IBAK0111
 471 CONTINUE                                                            IBAK0112
C  ORDER THE ITERATIONS
      CALL IORDER(N,KN1,ITER)                                            IBAK0113
```

```
C  RETURN TO THE MOST RECENT ITERATION DEFINED BY IORDER
  411 L4=L4+1                                                         IBAK0114
      KNT=ITER(L4)                                                    IBAK0115
C  IF ALL ITERATIONS HAVE BEEN EXAMINED NO SOLUTION EXISTS
      IF(KNT.LE.C) GO TO 450                                          IBAK0116
      AD=0.0                                                          IBAK0117
C  ASSIGN TO THE VARIABLES THE VALUES THAT WERE ASSIGNED IN ITERATIONS PREVIOUS
C   TO THE CHOSEN ITERATION
      LNT=KNT-1                                                       IBAK0118
      DO 475 J=1,NN                                                   IBAK0120
      ITSV(J)=0                                                       IBAKC121
      MV(J)=0                                                         IBAK0125
      DO 476 I=1,LNT                                                  IBAK0130
  476 IF(KOLSV(N+1,I).EQ.J) MV(J)=KOLSV(N+3,I)                        IBAK0132
  475 IF(MV(J).EQ.KL) ITSV(J)=N                                       IBAK0133
C  RETRIEVE THE VARIABLE USED IN CHOSEN  ITERATION.
      II=KOLSV(N+1,KNT)                                               IBAK0157
      IF(KOLSV(N+1,KNT).EQ.NN) GO TO 411                              IBAK0158
C  RECONSTRUCT CHOSEN ITERATION MATRIX(MAT)
      DO 425 I=1,NN                                                   IBAK0160
      NC(I)=0                                                         IBAK0165
  425 NZ(I)=0                                                         IBAK0170
      DO 420 J=1,NN                                                   IBAK0180
      DO 419 K=1,N                                                    IBAK0190
      MT(K,J)=IN(K,J)+KOLSV(K,LNT)                                    IBAK0210
      IF(MT(K,J).EQ.0) CALL KONFLT(NN,J,K,NZ,NC,IN,N)                 IBAK0220
  419 CONTINUE                                                        IBAK0224
      IF(ITSV(J).EQ.N) NZ(J)=N                                        IBAK0228
  420 CONTINUE                                                        IBAK0230
C  USE NEXT COLUMN WITH SAME NUMBER OF ZEROS, IF NO SUCH COLUMN EXISTS RETURN
C   TO NEXT FEASIBLE ITERATION
      IND=KOLSV(N+2,KNT)                                              IBAK0235
      CALL KVAR(NN,NZ,II,AD,MV,ITSV,NC,IND,NU,KNT,NX3)                IBAK0240
      KOLSV(N+2,KNT)=IND
      IF(AD.NE.0.0) GO TO 411                                         IBAK0250
C
C  DETERMINE THE NEW VALUE OF THE VARIABLE ,   INDICATE IF SOLUTION IS
C  INFEASIBLE SO THAT A NEW PATH CAN BE TAKEN, AND DEFINE THE NEW COLUMN
C   TO BE ADDED TO THE INITIAL TABLEAU
  430 CALL NZERO(N,NN,II,IN,ITSV,MV,E,KL,IA,BA,KM)                    IBAK0251
      IF(BA.NE.0.0) GO TO 430                                         IBAK0252
C  IF COLOR CHOSEN IS AT MAXIMUM VALUE, GO TO ANOTHER COLUMN
      IF(E.NE.0.0)CALL KLRMCH(N,NN,II,AD,MV,NZ,IN,ITSV,IA,E,MT,KL,KOLSV,IBAK0253
     1KNT,KM,NC,IND,NU,N3,NX3)                                        IBAK0254
C  IF NO OTHER COLUMN CAN BE USED, RETURN TO NEXT FEASIBLE ITERATION
      IF(AD.NE.0.0) GO TO 411                                         IBAK0255
C  SAVE NECESSARY VALUES TO BE USED IN RECONSTRUCTING THIS ITERATION IF REQUIRED
  441 DO 440 I=1,N                                                    IBAK0258
```

```
  440 KOLSV(I,KNT)=IA(I)                                                IBAK0280
      KOLSV(N+1,KNT)=II                                                 IBAK0290
      KOLSV(N+3,KNT)=MV(II)                                             IBAKC305
      RETURN                                                            IBAK0310
  450 D=NN                                                              IBAK0320
      WRITE(6,451)                                                      IBAK0330
  451 FORMAT(10X,'ALL COMBINATIONS HAVE BEEN TRIED, NO SOLUTION EXISTS')IBAK0340
      RETURN                                                            IBAK0350
      END


      SUBROUTINE KVAR(NN,MZ,II,AD,MV,ISV,NC,IND,NU,KNT,NX3)             KVAR0010
C SUBPROGRAM TO FIND THE NEXT COLUMN IN LINE WITH THE LEAST NUMBER OF ZEROS.
C  SYMBOLS ARE: MZ=NZ ; OTHERS AS PREVIOUSLY DEFINED
C
      DIMENSION MZ(NN),MV(NN),ISV(NN),NC(NN),NU(NX3)                    KVAR0020
C  ASSIGN INITIAL VALUES
      ISV(II)=0                                                         KVAR0030
      IF(NU(KNT).NE.0) GO TO 501                                        KVAR0035
      NP=NN-1                                                           KVAR0040
C  IF ALL COLUMNS WITH  NONBORDERING ZEROS HAVE BEEN EXAMINED, EXAMINE TOTAL
C   NUMBER OF ZEROS IN EACH COLUMN FOR NEXT COLUMN TO USE.
      IF(IND.NE.0) GO TO 540                                            KVAR0048
C  CHECK THE REST OF THE COLUMNS IN THE MATRIX FOR THE NEXT COLUMN WITH SAME
C  NUMBER OF  NONBORDERING ZEROS.
      DO 520 K=II,NP                                                    KVAR0050
      IF(NC(K+1).EQ.NC(II))  GO TO 530                                  KVAR0051
  520 CONTINUE                                                          KVAR0052
C  IF NO OTHER NONBORDERING ZERO COLUMN CAN BE CHOSEN, ASSIGN POSITIVE VALUE
C  TO INDICATOR
      IND=NN                                                            KVAR0053
      II=1                                                              KVAR0054
C  DETERMINE WHICH COLUMN HAS THE LEAST TOTAL NUMBER OF ZEROS
      DO 510 K=1,NP                                                     KVAR0057
  510 IF(MZ(K+1).LT.MZ(II)) II=K+1                                      KVAR0058
      RETURN                                                            KVAR0059
C  CHECK THE REST OF THE COLUMNS IN THE MATRIX FOR THE NEXT COLUMN WITH SAME
C  TOTAL NUMBER OF ZEROS.
  540 DO 500 K=II,NP                                                    KVAR0060
      IF(MZ(K+1).EQ.MZ(II)) GO TO 530                                   KVAR0070
  500 CONTINUE                                                          KVAR0080
C  IF NO OTHER COLUMN WITH SAME NUMBER OF ZEROS EXISTS, NOTE SAME AND RETURN
  501 AD=NN                                                             KVAR0090
      RETURN                                                            KVAR0100
C  DEFINE NEW VARIABLE TO BE USED
  530 II=K+1
      RETURN                                                            KVAR0130
      END
```

```
      SUBROUTINE INFEAS(N,NN,MV,IN,II,J,KL,IA,BA,KM,K2)                 INFS0010
C SUBROUTINE TO DETERMINE WHICH STATES BORDER ON STATE CHOSEN.   SYMBOLS ARE
C  AS PREVIOUSLY DEFINED
C
      DIMENSION MV(NN),IN(N,NN),IA(N),KM(NN)                            INFS0020
C  INCREASE COUNTER BY ONE; INDICATES WHEN ALL BORDERS HAVE BEEN CHECKED
      K2=K2+1                                                           NZER0033
      IF(II.EQ.1) GO TO 710                                             INFS0040
C  EXAMINE ROW FROM STATE#1 TO CHOSEN  STATE MINUS 1 TO SEE IF AN ADJACENT
C   STATE IS DEFINED IN THIS AREA
      IL=II-1                                                           INFS0050
      DO 720 KK=1,IL                                                    INFS0060
C  IF THERE EXISTS AN ADJACENT AREA, CHECK THE TWO COLORS FOR A MATCH
      IF(IN(J,KK).NE.0) CALL ICHK(N,NN,II,IN,MV,KK,KL,IA,BA,KM,K2)      INFS0070
      IF(BA.NE.0.0) RETURN                                              INFS0075
  720 CONTINUE                                                          INFS0090
      IF(II.EQ.NN) RETURN                                               INFS0100
C  EXAMINE ROW FROM CHOSEN  STATE PLUS 1 TO STATE#NN TO SEE IF AN ADJACENT
C   STATE IS DEFINED IN THIS AREA
  710 IU=II+1                                                           INFS0110
      DO 725 KK=IU,NN                                                   INFS0120
C  IF THERE EXISTS AN ADJACENT AREA, CHECK THE TWO COLORS FOR A MATCH
      IF(IN(J,KK).NE.0) CALL ICHK(N,NN,II,IN,MV,KK,KL,IA,BA,KM,K2)      INFS0130
      IF(BA.NE.0.0) RETURN                                              INFS0135
  725 CONTINUE                                                          INFS0150
      RETURN                                                            INFS0160
      END


      SUBROUTINE ICHK(N,NN,II,IN,MV,K,KL,IA,BA,KM,K2)                   ICHK0010
C SUBROUTINE COMPARE THE COLORS OF ADJACENT AREAS.  IF THE COLORS ARE THE SAME
C  THE ROUTINE RETURNS TO FIND ANOTHER COLOR OR VARIABLE IF NECESSARY.
C  WHEN THE COLOR HAS BEEN ESTABLISHED, THE NEW COLUMN TO ADD TO THE INITIAL
C  TABLEAU IS DEFINED.   SYMBOLS ARE AS PREVIOUSLY DEFINED.
C
      DIMENSION MV(NN),IA(N),IN(N,NN),KM(NN)                            ICHK0020
C
C  IF ADJACENT AREAS ARE THE SAME COLOR, RETURN TO STARTING POINT AND REDEFINE
C   COLOR OR VARIABLE
      IF(MV(II).EQ.MV(K)) GO TO 830                                     ICHK0040
C  WHEN ALL OF AN AREA'S BORDERS HAVE BEEN CHECKED, DEFINE THE NEW COLUMN TO
C   BE ADDED
      IF(KM(II).EQ.K2) GO TO 813                                        ICHK0056
C  IF ALL BORDER AREAS HAVE NOT BEEN CHECKED, RETURN TO SUBROUTINE INFEAS
C    AND CHECK NEXT BORDER AREA
      RETURN                                                            ICHK0057
```

```
C
  813 DO 820 J=1,N                                                     ICHK0068
  820 IA(J)=0                                                          ICHK0070
      DO 825 J=1,N                                                     ICHK0080
      DO 825 JJ=1,NN                                                   ICHK0090
      IA(J)=IA(J)+IN(J,JJ)*MV(JJ)                                      ICHK0100
  825 CONTINUE                                                         ICHK0110
      RETURN                                                           ICHK0120
  830 BA=N                                                             ICHK0130
      RETURN                                                           ICHK0140
      END

      SUBROUTINE KLRMCH(N,NN,II,AD,MV,NZ,IN,ITSV,IA,E,MT,KL,KSV,KT,KM,  KLMH0010
     1NC,IND,NU,N3,NX3)                                                KLMH0011
C  SUBROUTINE TO DEFINE ANOTHER COLUMN TO USE, IF THE SOLUTION IS INFEASIBLE
C    AND AT MAXIMUM COLOR VALUE.   SYMBOLS ARE:  AS DEFINED ELSEWHERE
C
      DIMENSION MV(NN),NZ(NN),IN(N,NN),ITSV(NN),IA(N),MT(N,NN),KM(NN)   KLMH0020
      DIMENSION NC(NN),KSV(N3,NX3),NU(NX3)                             KLMH0021
C
C  ZERO DUMMIES USED
  700 E=0.0                                                            KLMH0030
      BA=0.0                                                           KLMH0031
      KE=KT-1                                                          KLMH0040
C  DEFINE COLOR OF VARIABLE TO BE A PREVIOUS COLOR USED.
      MV(II)=0                                                         KLMH0050
      DO 740 I=1,KE                                                    KLMH0051
  740 IF(KSV(N+1,I).EQ.II) MV(II)=KSV(N+3,I)                           KLMH0052
C  USE THE NEXT COLUMN WITH SAME NUMBER OF ZEROS.  IF NO SUCH COLUMN EXISTS,
C    RETURN TO NEXT APPLICABLE ITERATION
      CALL KVAR(NN,NZ,II,AD,MV,ITSV,NC,IND,NU,KT,NX3)                  KLMH0060
C  SAVE THE VALUE OF THE INDICATOR
      KSV(N+2,KT)=IND                                                  KLMH0065
      IF(AD.NE.0.0) RETURN                                             KLMH0070
C  DETERMINE THE NEW VALUE OF THE VARIABLE ,   INDICATE IF SOLUTION IS
C  INFEASIBLE SO THAT A NEW PATH CAN BE TAKEN, AND DEFINE THE NEW COLUMN
C   TO BE ADDED TO THE INITIAL TABLEAU
  710 CALL NZERO(N,NN,II,IN,ITSV,MV,E,KL,IA,BA,KM)                     KLMH0080
      IF(BA.NE.0.0) GO TO 710                                          KLMH0085
      IF(E.NE.0.0) GO TO 700                                           KLMH0090
C  REZERO PRESENT INDICATOR
      IND=0                                                            KLMH0100
      RETURN                                                           KLMH0120
      END

      SUBROUTINE KONFLT(NN,J1,K1,NZ,NZCON,IN,N)                        KNFT0010
```

```
C   SUBPROGRAM TO COUNT THE NUMBER OF ZEROS IN EACH COLUMN AND TO COUNT THE
C     NUMBER OF ZEROS ASSOCIATED WITH NONBORDERING AREAS.  SYMBOLS ARE AS
C     PREVIOUSLY DEFINED
C
      DIMENSION NZ(NN),NZCON(NN),IN(N,NN)                             KNFT0020
      NZ(J1)=NZ(J1)+1                                                  KNFT0030
      IF(IN(K1,J1).EQ.0) NZCON(J1)=NZCON(J1)+1                         KNFT0040
      RETURN                                                           KNFT0050
      END


      SUBROUTINE NOMCH(N,NN,IIS,II,KK,MV,IA,IN,NU,KNT,KL,ITSV,NX3)     NMCH0010
C   IN THE EVENT THAT THE COLOR CHOSEN  WHEN ENUMERATING ALL POSSIBLE COLORS
C     FOR THE CHOSEN AREA DOES NOT MATCH ANY OF THE BORDERING AREAS,THIS
C     SUBPROGRAM DEFINES THE VARIABLE, ITS COLOR, AND THE NEW COLUMN TO ADD
C
      DIMENSION MV(NN),IA(N),IN(N,NN),ITSV(NN),NU(NX3)                 NMCH0020
C   DEFINE CHCSEN  VARIABLE
      II=IIS                                                           NMCH0030
C   DEFINE VARIABLE'S COLOR
      MV(II)=KK                                                        NMCH0040
C   IF COLOR IS AT THE MAXIMUM VALUE, MAKE A NOTE OF THE VARIABLE
      IF(MV(II).EQ.KL) ITSV(II)=N                                      NMCH0050
C   DEFINE THE COLUMN TO ADD
      DO 1200 J=1,N                                                    NMCH0060
 1200 IA(J)=0                                                          NMCH0070
      DO 1251 J=1,N                                                    NMCH0080
      DO 1250 JJ=1,NN                                                  NMCH0090
 1250 IA(J)=IA(J)+IN(J,JJ)*MV(JJ)                                      NMCH0100
 1251 CONTINUE                                                         NMCH0110
      KNT=KNT+1                                                        NMCH0115
      NU(KNT)=N                                                        NMCH0118
      RETURN                                                           NMCH0120
      END


      SUBROUTINE IORDER(N,KN1,ITER)                                    IORD0010
C   SUBPROGRAM TO ORDER THE ITERATIONS CHOSEN  FOR BACKTRACKING
C     SYMBOLS ARE AS DEFINED IN SUBROUTINE IBAK
C
      DIMENSION ITER(N)                                                IORD0020
      KN1M1=KN1-1                                                      IORD0030
      DO 1010 J=1,KN1M1                                                IORD0035
      IG=ITER(J)                                                       IORD0036
      J1=J+1                                                           IORD0037
      DO 1011 I=J1,KN1                                                 IORD0038
      IF(IG.GE.ITER(I)) GO TO 1011                                     IORD0039
      IG=ITER(I)                                                       IORD0040
```

```
      ITER(I)=ITER(J)                                                   IORD0041
      ITER(J)=IG                                                        IORD0042
 1011 CONTINUE                                                          IORD0043
 1010 CONTINUE                                                          IORD0070
      RETURN                                                            IORD0080
      END


      SUBROUTINE LKINF(N,NN,IIS,KN1,IN,MV,MVMCH,JJ)                     LKIF0010
C  SUBROUTINE TO DETERMINE WHICH STATES BORDER ON CHOSEN  STATE.  SYMBOLS
C   ARE AS PREVIOUSLY DEFINED
C
      DIMENSION IN(N,NN),MV(NN),MVMCH(NN)                               LKIF0020
      IF(IIS.EQ.1) GO TO 910                                            LKIF0030
C  EXAMINE ROW FROM STATE #1 TO CHOSEN  STATE MINUS 1 TO SEE IF AN ADJACENT
C   STATE IS DEFINED IN THIS AREA.
      ISL=IIS-1                                                         LKIF0040
      DO 920 KK=1,ISL                                                   LKIF0050
C  IF THERE EXISTS AN ADJACENT AREA, CHECK THE TWO COLORS FOR A MATCH
      IF(IN(JJ,KK).NE.0) CALL LKICHK(NN,IIS,KK,KN1,MV,MVMCH)            LKIF0060
  920 CONTINUE                                                          LKIF0070
      IF(IIS.EQ.NN) RETURN                                              LKIF0080
C  EXAMINE ROW FROM CHOSEN  STATE PLUS 1 TO STATE #NN TO SEE IF AN ADJACENT
C   STATE IS DEFINED IN THIS AREA.
  910 ISU=IIS+1                                                         LKIF0090
      DO 925 KK=ISU,NN                                                  LKIF0100
C  IF THERE EXISTS AN ADJACENT AREA, CHECK THE TWO COLORS FOR A MATCH
      IF(IN(JJ,KK).NE.0) CALL LKICHK(NN,IIS,KK,KN1,MV,MVMCH)            LKIF0110
  925 CONTINUE                                                          LKIF0120
      RETURN                                                            LKIF0130
      END


      SUBROUTINE LKICHK(NN,IIS,K,KN1,MV,MVMCH)                          LKIK0010
C  SUBROUTINE COMPARES THE COLORS OF ADJACENT AREAS.  IF THE COLORS ARE THE
C   SAME, THE ROUTINE RECORDS WHICH AREA HAS SAME COLOR AS CHOSEN
C   AREA.   SYMBOLS ARE AS DEFINED IN IBAK
C
      DIMENSION MV(NN),MVMCH(NN)                                        LKIK0020
C  COMPARE THE COLORS OF THE ADJACENT AREAS
      IF(MV(IIS).EQ.MV(K)) GO TO 890                                    LKIK0030
      RETURN                                                            LKIK0040
C  RECORD THE VARIABLE WITH THE MATCHING COLOR
  890 KN1=KN1+1                                                         LKIK0044
      MVMCH(KN1)=K                                                      LKIK0045
      RETURN                                                            LKIK0050
      END

      SUBROUTINE KNTVAR(N,NN,IN,I,IITMP,L2)                             KNVR0010
```

```
C   SUBPROGRAM TO DETERMINE WHICH TWO VARIABLES ARE ACJACENT AND STILL HAVE
C    COLOR ZERO.  SYMBOLS ARE AS DEFINED IN IBAK
C
      DIMENSION IN(N,NN),IITMP(NN)                                      KNVR0020
      DO 13C0 J=1,NN                                                    KNVR0030
      IF(IN(I,J).NE.0) L2=L2+1                                          KNVR0040
 1300 IF(IN(I,J).NE.0) IITMP(L2)=J                                      KNVR0050
      RETURN                                                            KNVR0060
      END
```

BIBLIOGRAPHY

Courant, R. and Robbins, H., What Is Mathematics?, p. 246-248, Oxford University Press, 1941.

Dantzig, G. B., "On Significance of Solving L. P. Problem with Some Integer Variables," Econometrica, V. 28, p. 30-44, January 1960.

Greenberg, H., "An Algorithm for the Computation of Knapsack Function," Journal of Mathematical Analysis and Applications, v. 26, No. 1, p. 159-162, April 1969.

Greenberg, H., "A Dynamic Programming Solution to the Integer Linear Program," Journal of Mathematical Analysis and Applications, v. 26, No. 2, p. 454-459, May 1969.

Oystein, O., The Four-Color Problem, Academic Press, 1967.

Waller, B. E., III, The Color Problem, M. A. Thesis, Vanderbilt University, Nashville, 1965.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3.	Chief of Naval Operations OP-96 Department of the Navy Washington, D. C. 20350	1
4.	Asst. Professor H. Greenberg, Code 55Gd (thesis advisor) Department of Operations Analysis Naval Postgraduate School Monterey, California 93940	1
5.	LCDR Dennis S. Read, USN (student) USS STERETT (DLG 31) c/o F.P.O. San Francisco, Calif. 96601	1

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified 2b. GROUP

3. REPORT TITLE

An Iterative Process to Solve the Graph-Coloring Problem

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*

Master's Thesis, October 1969

5. AUTHOR(S) *(First name, middle initial, last name)*

Dennis S. Read

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
October 1969	39	6

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
b. PROJECT NO.	
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	

10. DISTRIBUTION STATEMENT

This document has been approved for public release and sale; its distribution is unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Naval Postgraduate School Monterey, California 93940

13. ABSTRACT

The intent of this paper is to describe a method of coloring a map and to present an algorithm for the solution of this problem. A computer program was developed to provide solutions to the problem of coloring a map which consists of a finite number of areas. This algorithm may also be applied to problems other than map-coloring.

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Iterative process						
Map-coloring						
Four color problem						